

# Horizon automation

*From 101 to deepdive*



VMware {code}



**VMUG**  
VMWARE USER GROUP

Your Link to the VMware Community

## Disclaimer

Everything in this presentation is allowed to be stolen, borrowed or re-used without the presenters consent. There is no penalty for that but they won't say no to an appreciative alcoholic beverage (if they have to drive preferably in a closed bottle) when we are able to meet in person. It is allowed to make screenshots but it could be just as handy to ask for the slide deck or the recording. Good and bad ideas are always welcome and could be added to future presentations if the presenters wants to do so. Bugs, faults and errors are not to blame on the presenter, but the manuals or network. Please provide feedback about them to the presenter. The presenter is neither responsible nor accountable for complicated or annoying use of phrases, bad jokes or harsh language. Any feedback is welcome and can be given through Twitter, vmtn, Slack, Reddit, Zoom, email, postal pigeon or by word of mouth.

# Introductie

```
$Wouter=New-Object PSObject -Property @{
```

```
    "Name"           = "Wouter Kursten";
```

```
    "Employer"       = "ControlUP";
```

```
    "Job"            = "Professional Services Engineer";
```

```
    "Certifications" = "VCIX-DCV, VCIX-DTM, VCP-  
                        NV etc etc";
```

```
    "vCommunity"     = "vExpert (EUC), VMware EUC Champion,  
VMware{Code} CodeCoach, Nutanix Technology Champion";
```

```
    "Blog"           = https://www.retouw.nl;
```

```
    "Twitter"        = "@Magneet_NL";
```

```
    "E-Mail"         = "Wouter@Retouw.NL";
```

```
    "Github"         = "https://github.com/Magneet"
```

```
}
```



# A Brief History of Horizon and PowerShell

---

- **First PowerShell Module for Horizon was released back in 2010 with View 4.5**
- **Some short-lived community attempts at creating custom View PowerShell Modules**
- **SOAP API method introduced with PowerCLI 6.5**
- **Rest API method introduced with 7.10**

- **Raw API Access**
- **Installed and Managed through PowerShell Gallery with other PowerCLI Modules**
  - Install-Module VMware.PowerCLI

## Requirements:

- Horizon 7.02 or Newer
- PowerShell 5
- Does not work with PowerShell Core

# Why Automate Horizon?

---

- **Managing Horizon can be a time-consuming task**
- **Many tasks need to happen on a recurring basis**
  - Thanks, Microsoft!

**Does not scale well as environments grow larger and more complex**

# Getting Started with Horizon PowerCLI

- **Installing PowerCLI**
  - Install-Module VMware.PowerCLI
  - Includes all supported PowerCLI Modules
- **No need to import the module – it will load when running one of the cmdlets**

```
Administrator: Windows PowerShell

PS C:\Windows\system32> install-module vmware.powercli

Installing package 'VMware.PowerCLI'
Installing dependent package 'VMware.VimAutomation.Core'
[ooooooooooooooooooooo
Installing package 'VMware.VimAutomation.Core'
Downloaded 39907763,00 MB out of 39907763,00 MB.
[ooooooooooooooooooooo

Untrusted repository
You are installing the modules from an untrusted repository. If you trust this repository,
InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure you want to i
'PSGallery'?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): a
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\DemoUser> connect-hvserver pod1cbr1
WARNING: Please consider joining the VMware Customer Experience Improvement
better product. You can join using the following command:

Set-PowerCLIConfiguration -Scope User -ParticipateInCEIP $true

VMware's Customer Experience Improvement Program ("CEIP") provides VMware w
improve its products and services, to fix problems, and to advise you on ho
part of the CEIP, VMware collects technical information about your organiza
on a regular basis in association with your organization's VMware license k
identify any individual.

For more details: type "help about_ceip" to see the related help article.

To disable this warning and set your preference use the following command a
Set-PowerCLIConfiguration -Scope User -ParticipateInCEIP $true or $false.
```

# Connecting to a Horizon Environment

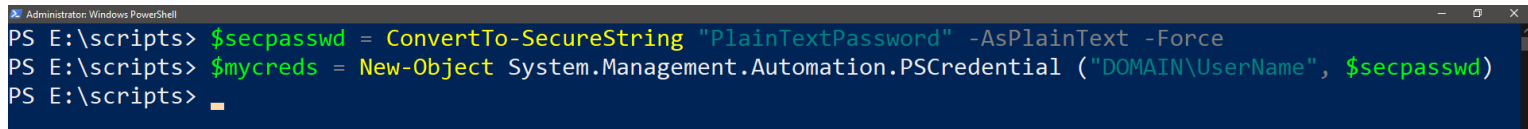
- **Connection command: Connect-HVServer**

- **Store this as a variable**

- Makes it easier to work with

- **Common Parameters**

- Server: Horizon Connection Server Address
- Credential: PSCredential Object
  - Include Username and Domain



```
Administrator: Windows PowerShell
PS E:\scripts> $secpasswd = ConvertTo-SecureString "PlainTextPassword" -AsPlainText -Force
PS E:\scripts> $mycreds = New-Object System.Management.Automation.PSCredential ("DOMAIN\UserName", $secpasswd)
PS E:\scripts>
```

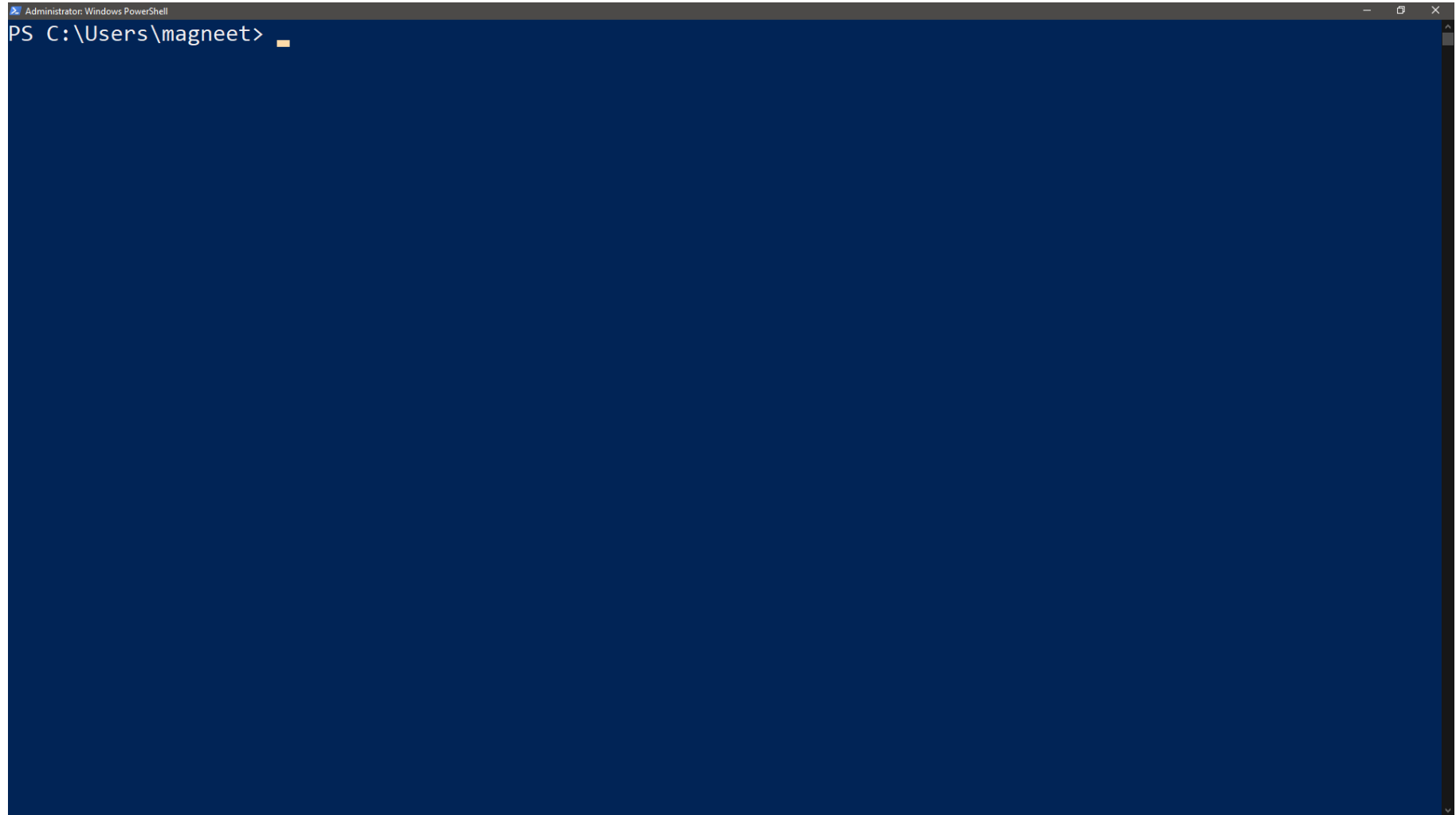
- **Timeout**

- Same timeout as administrator sessions



# Connecting to a Horizon Environment

---



# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
```

# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
```

# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> █
```

# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secpasswd)
```

# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secpasswd)
PS C:\Users\magneet>
```

# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secpasswd)
PS C:\Users\magneet> $hvserver=Connect-HVServer pod1cbr1.magneet.lab -Credential $mycreds
```

# Connecting to a Horizon Environment

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secpasswd)
PS C:\Users\magneet> $hvserver=Connect-HVServer pod1cbr1.magneet.lab -Credential $mycreds
PS C:\Users\magneet> █
```



# HVServer Extension Data

---

- **Horizon API Interactions are handled through the HVServer Object Extension Data**
- **Extension Data is a collection of services and their methods, queries, and data stored in PSObjects that covers the entire set of management functions for Horizon**
  - May include nested objects

## Retrieving Extension Data

- `$HVServer.ExtensionData` – Shows a list of services to work with
- Store in variable to make it easier to work with in scripts
  - `$hvservices=$hvserver.extensiondata`
- Services can be accessed by appending them to the extension data
  - `$hvservices.ConnectionServerHealth`
- Use Get-Member to explore the methods and queries of each Service

# HVServer Extension Data

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secp
asswd)
PS C:\Users\magneet> $hvserver=Connect-HVServer pod1cbr1.magneet.lab -Credential $mycreds
PS C:\Users\magneet> $hvservices=$hvserver.ExtensionData
```

# HVServer Extension Data

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secp
asswd)
PS C:\Users\magneet> $hvserver=Connect-HVServer pod1cbr1.magneet.lab -Credential $mycreds
PS C:\Users\magneet> $hvservices=$hvserver.ExtensionData
PS C:\Users\magneet>
```

# HVServer Extension Data

```
Administrator: Windows PowerShell
PS C:\Users\magneet> $secpasswd=read-host -AsSecureString
*****
PS C:\Users\magneet> $mycreds = New-Object System.Management.Automation.PSCredential ("magneet\m_wouter", $secp
asswd)
PS C:\Users\magneet> $hvserver=Connect-HVServer pod1cbr1.magneet.lab -Credential $mycreds
PS C:\Users\magneet> $hvservices=$hvserver.ExtensionData
PS C:\Users\magneet> $hvservices_
```

# HVServer Extension Data

```
Administrator: Windows PowerShell
Session : VMware.Hv.Session
SessionStatistics : VMware.Hv.SessionStatistics
Site : VMware.Hv.Site
SpaceReclamation : VMware.Hv.SpaceReclamation
StorageAccelerator : VMware.Hv.StorageAccelerator
StorageAcceleratorHost : VMware.Hv.StorageAcceleratorHost
Syslog : VMware.Hv.Syslog
Task : VMware.Hv.Task
URLRedirection : VMware.Hv.URLRedirection
UnauthenticatedAccessUser : VMware.Hv.UnauthenticatedAccessUser
UsageStatistics : VMware.Hv.UsageStatistics
UserEntitlement : VMware.Hv.UserEntitlement
UserHomeSite : VMware.Hv.UserHomeSite
Validator : VMware.Hv.Validator
ViewClient : VMware.Hv.ViewClient
ViewComposerDomainAdministrator : VMware.Hv.ViewComposerDomainAdministrator
ViewComposerHealth : VMware.Hv.ViewComposerHealth
VirtualCenter : VMware.Hv.VirtualCenter
VirtualCenterHealth : VMware.Hv.VirtualCenterHealth
VirtualCenterStatistics : VMware.Hv.VirtualCenterStatistics
VirtualDisk : VMware.Hv.VirtualDisk
VirtualMachine : VMware.Hv.VirtualMachine
VirtualSAN : VMware.Hv.VirtualSAN
VmFolder : VMware.Hv.VmFolder
VmTemplate : VMware.Hv.VmTemplate

PS C:\Users\magneet>
```

# HVServer Extension Data

```
Administrator: Windows PowerShell
ResourcePool                : VMware.Hv.ResourcePool
ResourceSettings            : VMware.Hv.ResourceSettings
Role                        : VMware.Hv.Role
SAMLAuthenticator           : VMware.Hv.SAMLAuthenticator
SAMLAuthenticatorHealth     : VMware.Hv.SAMLAuthenticatorHealth
SecondaryCredentials        : VMware.Hv.SecondaryCredentials
SecurityServer              : VMware.Hv.SecurityServer
SecurityServerHealth        : VMware.Hv.SecurityServerHealth
Session                    : VMware.Hv.Session
SessionStatistics           : VMware.Hv.SessionStatistics
Site                       : VMware.Hv.Site
SpaceReclamation            : VMware.Hv.SpaceReclamation
StorageAccelerator          : VMware.Hv.StorageAccelerator
StorageAcceleratorHost      : VMware.Hv.StorageAcceleratorHost
Syslog                     : VMware.Hv.Syslog
Task                       : VMware.Hv.Task
URLRedirection              : VMware.Hv.URLRedirection
UnauthenticatedAccessUser   : VMware.Hv.UnauthenticatedAccessUser
UsageStatistics             : VMware.Hv.UsageStatistics
UserEntitlement             : VMware.Hv.UserEntitlement
UserHomeSite               : VMware.Hv.UserHomeSite
Validator                  : VMware.Hv.Validator
ViewClient                 : VMware.Hv.ViewClient
ViewComposerDomainAdministrator : VMware.Hv.ViewComposerDomainAdministrator
ViewComposerHealth         : VMware.Hv.ViewComposerHealth
VirtualCenter              : VMware.Hv.VirtualCenter
VirtualCenterHealth        : VMware.Hv.VirtualCenterHealth
VirtualCenterStatistics     : VMware.Hv.VirtualCenterStatistics
VirtualDisk                : VMware.Hv.VirtualDisk
VirtualMachine             : VMware.Hv.VirtualMachine
VirtualSAN                 : VMware.Hv.VirtualSAN
VmFolder                   : VMware.Hv.VmFolder
VmTemplate                 : VMware.Hv.VmTemplate

PS D:\homelab> $hvservices.GatewayHealth | get-member
```

# HVServer Extension Data

```
Administrator: Windows PowerShell
UnauthenticatedAccessUser      : VMware.Hv.UnauthenticatedAccessUser
UsageStatistics                 : VMware.Hv.UsageStatistics
UserEntitlement                 : VMware.Hv.UserEntitlement
UserHomeSite                   : VMware.Hv.UserHomeSite
Validator                      : VMware.Hv.Validator
ViewClient                     : VMware.Hv.ViewClient
ViewComposerDomainAdministrator : VMware.Hv.ViewComposerDomainAdministrator
ViewComposerHealth             : VMware.Hv.ViewComposerHealth
VirtualCenter                  : VMware.Hv.VirtualCenter
VirtualCenterHealth            : VMware.Hv.VirtualCenterHealth
VirtualCenterStatistics        : VMware.Hv.VirtualCenterStatistics
VirtualDisk                    : VMware.Hv.VirtualDisk
VirtualMachine                 : VMware.Hv.VirtualMachine
VirtualSAN                     : VMware.Hv.VirtualSAN
VmFolder                       : VMware.Hv.VmFolder
VmTemplate                     : VMware.Hv.VmTemplate

PS D:\homelab> $hvservices.GatewayHealth | get-member

TypeName: VMware.Hv.GatewayHealth

Name      MemberType Definition
-----
Equals    Method      bool Equals(System.Object obj)
GatewayHealth_Get Method      VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
GatewayHealth_List Method      VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
GetHashCode Method      int GetHashCode()
GetType   Method      type GetType()
ToString  Method      string ToString()
Client    Property    VMware.Hv.HviClient Client {get;}
MoRef     Property    VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab>
```

- **Basically functions built into the API**
- **Allow you to perform Get, Create, Update and Delete operations against the API Services**
- **Uses PSObjects heavily**
- **Some methods require specific identifiers to retrieve data about an object - ie Desktop Pools**
  - This data must be queried or retrieved using different methods
  - ID's are Horizon-specific ID's, not vCenter Object ID's.



# Methods Example

```
Administrator: Windows PowerShell
UnauthenticatedAccessUser      : VMware.Hv.UnauthenticatedAccessUser
UsageStatistics                 : VMware.Hv.UsageStatistics
UserEntitlement                 : VMware.Hv.UserEntitlement
UserHomeSite                   : VMware.Hv.UserHomeSite
Validator                      : VMware.Hv.Validator
ViewClient                     : VMware.Hv.ViewClient
ViewComposerDomainAdministrator : VMware.Hv.ViewComposerDomainAdministrator
ViewComposerHealth             : VMware.Hv.ViewComposerHealth
VirtualCenter                  : VMware.Hv.VirtualCenter
VirtualCenterHealth            : VMware.Hv.VirtualCenterHealth
VirtualCenterStatistics        : VMware.Hv.VirtualCenterStatistics
VirtualDisk                    : VMware.Hv.VirtualDisk
VirtualMachine                  : VMware.Hv.VirtualMachine
VirtualSAN                     : VMware.Hv.VirtualSAN
VmFolder                       : VMware.Hv.VmFolder
VmTemplate                     : VMware.Hv.VmTemplate

PS D:\homelab> $hvservices.GatewayHealth | get-member

TypeName: VMware.Hv.GatewayHealth

Name      MemberType Definition
-----
Equals    Method      bool Equals(System.Object obj)
GatewayHealth_Get Method      VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
GatewayHealth_List Method      VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
GetHashCode Method      int GetHashCode()
GetType   Method      type GetType()
ToString  Method      string ToString()
Client    Property    VMware.Hv.HviClient Client {get;}
MoRef     Property    VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get_
```

# Methods Example

```
Administrator: Windows PowerShell
VirtualCenter           : VMware.Hv.VirtualCenter
VirtualCenterHealth     : VMware.Hv.VirtualCenterHealth
VirtualCenterStatistics : VMware.Hv.VirtualCenterStatistics
VirtualDisk             : VMware.Hv.VirtualDisk
VirtualMachine          : VMware.Hv.VirtualMachine
VirtualSAN              : VMware.Hv.VirtualSAN
VmFolder               : VMware.Hv.VmFolder
VmTemplate              : VMware.Hv.VmTemplate

PS D:\homelab> $hvservices.GatewayHealth | get-member

    TypeName: VMware.Hv.GatewayHealth

Name      MemberType Definition
-----
Equals    Method      bool Equals(System.Object obj)
GatewayHealth_Get Method      VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
GatewayHealth_List Method      VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
GetHashCode Method      int GetHashCode()
GetType   Method      type GetType()
ToString  Method      string ToString()
Client    Property    VMware.Hv.HviClient Client {get;}
MoRef     Property    VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)

PS D:\homelab>
```

# Methods Example

```
Administrator: Windows PowerShell
VirtualCenter           : VMware.Hv.VirtualCenter
VirtualCenterHealth     : VMware.Hv.VirtualCenterHealth
VirtualCenterStatistics : VMware.Hv.VirtualCenterStatistics
VirtualDisk             : VMware.Hv.VirtualDisk
VirtualMachine          : VMware.Hv.VirtualMachine
VirtualSAN              : VMware.Hv.VirtualSAN
VmFolder                : VMware.Hv.VmFolder
VmTemplate              : VMware.Hv.VmTemplate

PS D:\homelab> $hvservices.GatewayHealth | get-member

TypeName: VMware.Hv.GatewayHealth

Name      MemberType Definition
-----
Equals    Method      bool Equals(System.Object obj)
GatewayHealth_Get Method      VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
GatewayHealth_List Method      VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
GetHashCode Method      int GetHashCode()
GetType   Method      type GetType()
ToString  Method      string ToString()
Client    Property    VMware.Hv.HviClient Client {get;}
MoRef     Property    VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list
```

# Methods Example

```
Administrator: Windows PowerShell

PS D:\homelab> $hvservices.GatewayHealth | get-member

TypeName: VMware.Hv.GatewayHealth

Name            MemberType Definition
----
Equals          Method      bool Equals(System.Object obj)
GatewayHealth_Get Method      VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
GatewayHealth_List Method      VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
GetHashCode     Method      int GetHashCode()
GetType         Method      type GetType()
ToString        Method      string ToString()
Client          Property    VMware.Hv.HviClient Client {get;}
MoRef           Property    VMware.Hv.ManagedObjectReference MoRef {get;}
```

```
PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
```

```
PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list_

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
```

```
PS D:\homelab>
```

# Methods Example

```
Administrator: Windows PowerShell

PS D:\homelab> $hvservices.GatewayHealth | get-member

TypeName: VMware.Hv.GatewayHealth

Name      MemberType Definition
-----
Equals     Method      bool Equals(System.Object obj)
GatewayHealth_Get Method      VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)
GatewayHealth_List Method      VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()
GetHashCode Method      int GetHashCode()
GetType    Method      type GetType()
ToString   Method      string ToString()
Client     Property    VMware.Hv.HvciClient Client {get;}
MoRef      Property    VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list_

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list()_
```

# Methods Example

```
Administrator: Windows PowerShell

Client      Property  VMware.Hv.HviClient Client {get;}
MoRef       Property  VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list_

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list()

Id           : VMware.Hv.GatewayId
Name          : podluag1
Address       : 10.101.0.105
GatewayZoneInternal : False
Version       : 3.8
Type          : AP
ConnectionData : VMware.Hv.GatewayHealthConnectionData
GatewayStatusActive : True
GatewayStatusStale  : False
GatewayContacted   : True

PS D:\homelab>
```

# Methods Example

```
Administrator: Windows PowerShell
Client      Property  VMware.Hv.HviClient Client {get;}
MoRef       Property  VMware.Hv.ManagedObjectReference MoRef {get;}

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_Get

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list

OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list()

Id           : VMware.Hv.GatewayId
Name          : pod1uag1
Address       : 10.101.0.105
GatewayZoneInternal : False
Version       : 3.8
Type          : AP
ConnectionData : VMware.Hv.GatewayHealthConnectionData
GatewayStatusActive : True
GatewayStatusStale  : False
GatewayContacted   : True

PS D:\homelab> ($hvservices.GatewayHealth.GatewayHealth_list()).ConnectionData
```

# Methods Example

```
Administrator: Windows PowerShell

-----
VMware.Hv.GatewayHealthInfo GatewayHealth_Get(VMware.Hv.GatewayId id)

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list_
OverloadDefinitions
-----
VMware.Hv.GatewayHealthInfo[] GatewayHealth_List()

PS D:\homelab> $hvservices.GatewayHealth.GatewayHealth_list()

Id                : VMware.Hv.GatewayId
Name              : pod1uag1
Address           : 10.101.0.105
GatewayZoneInternal : False
Version           : 3.8
Type              : AP
ConnectionData    : VMware.Hv.GatewayHealthConnectionData
GatewayStatusActive : True
GatewayStatusStale : False
GatewayContacted  : True

PS D:\homelab> ($hvservices.GatewayHealth.GatewayHealth_list()).ConnectionData_
NumActiveConnections NumPcoipConnections NumBlastConnections
-----
1                    0                    1

PS D:\homelab>
```



- **Queries are a way to retrieve and filter specific data from horizon**

- **Example:**

- `$queryservice = new-object vmware.hv.queryserviceservice`
- `$defn = new-object vmware.hv.querydefinition`
- `$defn.queryentitytype = 'MachineDetailView'`
- `$results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)`
- `$results.results`

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $queryservice = new-object vmware.hv.queryserviceservice
PS D:\homelab> $defn = new-object vmware.hv.querydefinition
PS D:\homelab> $defn.queryentitytype = 'MachineDetailsView'
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab>
```

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $queryservice = new-object vmware.hv.queryserviceservice
PS D:\homelab> $defn = new-object vmware.hv.querydefinition
PS D:\homelab> $defn.queryentitytype = 'MachineDetailsView'
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> $results
```

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $queryservice = new-object vmware.hv.queryserviceservice
PS D:\homelab> $defn = new-object vmware.hv.querydefinition
PS D:\homelab> $defn.queryentitytype = 'MachineDetailsView'
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> $results

Id                StartingOffset RemainingCount Results
--                -
VMware.Hv.QueryId 0                0 {VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId...}

PS D:\homelab>
```

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $queryservice = new-object vmware.hv.queryserviceservice
PS D:\homelab> $defn = new-object vmware.hv.querydefinition
PS D:\homelab> $defn.queryentitytype = 'MachineDetailsView'
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> $results

Id                StartingOffset RemainingCount Results
--                -
VMware.Hv.QueryId 0                0 {VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId...}

PS D:\homelab> $results.Results | Select-Object -first 1
```

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $queryservice = new-object vmware.hv.queryserviceservice
PS D:\homelab> $defn = new-object vmware.hv.querydefinition
PS D:\homelab> $defn.queryentitytype = 'MachineDetailsView'
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> $results

Id                StartingOffset RemainingCount Results
--                -
VMware.Hv.QueryId 0                0 {VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId...}

PS D:\homelab> $results.Results | Select-Object -first 1

Id                : VMware.Hv.MachineId
GroupId           :
Data              : VMware.Hv.MachineData
DesktopData       : VMware.Hv.MachineDesktopData
SessionData       :
ManagedMachineDetailsData : VMware.Hv.ManagedMachineDetailsData
MachineAgentPairingData    : VMware.Hv.MachineAgentPairingData

PS D:\homelab> 
```

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $queryservice = new-object vmware.hv.queryserviceservice
PS D:\homelab> $defn = new-object vmware.hv.querydefinition
PS D:\homelab> $defn.queryentitytype = 'MachineDetailsView'
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> $results

Id                StartingOffset RemainingCount Results
--                -
VMware.Hv.QueryId 0                0 {VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId...}

PS D:\homelab> $results.Results | Select-Object -first 1

Id                : VMware.Hv.MachineId
GroupId           :
Data              : VMware.Hv.MachineData
DesktopData       : VMware.Hv.MachineDesktopData
SessionData       :
ManagedMachineDetailsData : VMware.Hv.ManagedMachineDetailsData
MachineAgentPairingData  : VMware.Hv.MachineAgentPairingData

PS D:\homelab> ($results.results |select -first 1).data_
```

# Queries Example

```
Administrator: Windows PowerShell
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> $results

Id                StartingOffset RemainingCount Results
--
VMware.Hv.QueryId 0                0 {VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId, VMware.Hv.MachineId...}

PS D:\homelab> $results.Results | Select-Object -first 1

Id                : VMware.Hv.MachineId
GroupId           :
Data              : VMware.Hv.MachineData
DesktopData       : VMware.Hv.MachineDesktopData
SessionData       :
ManagedMachineDetailsData : VMware.Hv.ManagedMachineDetailsData
MachineAgentPairingData   : VMware.Hv.MachineAgentPairingData

PS D:\homelab> ($results.results |select -first 1).data

Name           : Pod01-4001
DnsName         : pod01-4001.loft.lab
Type            : MANAGED_VIRTUAL_MACHINE
AgentVersion    : 7.11.0
AccessGroup     : VMware.Hv.AccessGroupId
BasicState      : AVAILABLE
AssignedUser    :
AssignedUserName :
AgentBuildNumber : 15238678

PS D:\homelab>
```



# Queries Example

```
Administrator: Windows PowerShell
Id                : VMware.Hv.MachineId
GroupId           :
Data              : VMware.Hv.MachineData
DesktopData       : VMware.Hv.MachineDesktopData
SessionData       :
ManagedMachineDetailsData : VMware.Hv.ManagedMachineDetailsData
MachineAgentPairingData : VMware.Hv.MachineAgentPairingData
```

```
PS D:\homelab> ($results.results |select -first 1).data
```

```
Name             : Pod01-4001
DnsName           : pod01-4001.loft.lab
Type              : MANAGED_VIRTUAL_MACHINE
AgentVersion      : 7.11.0
AccessGroup       : VMware.Hv.AccessGroupId
BasicState        : AVAILABLE
AssignedUser      :
AssignedUserName   :
AgentBuildNumber  : 15238678
```

```
PS D:\homelab> ($results.results |select -first 1).DesktopData
```

```
Id                : VMware.Hv.DesktopId
Name              : Pod01_Pool04
Type              : AUTOMATED
Source            : INSTANT_CLONE_ENGINE
UserAssignment    : FLOATING
```

```
PS D:\homelab>
```

# Queries Example

```
Administrator: Windows PowerShell

PS D:\homelab> ($results.results | select -first 1).DesktopData

Id           : VMware.Hv.DesktopId
Name          : Pod01_Pool04
Type          : AUTOMATED
Source        : INSTANT_CLONE_ENGINE
UserAssignment : FLOATING

PS D:\homelab> ($results.results | select -first 1).ManagedMachineDetailsData_

VirtualCenter      : VMware.Hv.VirtualCenterId
HostName           : 10.101.0.25
DatastorePaths      : {NVME1TB}
VirtualDisks        : {VMware.Hv.MachineVirtualDiskData}
PersistentDisks     :
LastMaintenanceTime : 3/3/2020 12:00:22 PM
Operation           :
OperationState      : COMPLETED
AutoRefreshLogOffSetting :
InHoldCustomization : False
MissingInVCenter    : False
CreateTime          : 3/3/2020 12:00:12 PM
CloneErrorMessage   :
CloneErrorTime      :
BaseImagePath       : /Datacenter_Pod1/vm/w10-1903-IC
BaseImageSnapshotPath : /VM Snapshot 3%25252f3%25252f2020, 8:22:38 AM
PendingBaseImagePath :
PendingBaseImageSnapshotPath :
```

# Queries Example

---

## ■ Filtering

- `$defn.filter = New-Object VMware.Hv.QueryFilterContains -Property @{'memberName'='data.basicState'; 'value' = 'CONNECTED'}`

## ■ Multiple filter object types

- QueryFilter
- QueryFilterAnd
- QueryFilterBetween
- QueryFilterContains
- QueryFilterEquals
- QueryFilterNot
- QueryFilterNotEquals
- QueryFilterOr
- QueryFilterStartsWith

# Queries Example

```
Administrator: Windows PowerShell

PS D:\homelab> ($results.Results | select -first 1).DesktopData

Id           : VMware.Hv.DesktopId
Name          : Pod01_Pool04
Type          : AUTOMATED
Source        : INSTANT_CLONE_ENGINE
UserAssignment : FLOATING

PS D:\homelab> ($results.Results | select -first 1).ManagedMachineDetailsData_

VirtualCenter      : VMware.Hv.VirtualCenterId
HostName           : 10.101.0.25
DatastorePaths      : {NVME1TB}
VirtualDisks        : {VMware.Hv.MachineVirtualDiskData}
PersistentDisks     :
LastMaintenanceTime : 3/3/2020 12:00:22 PM
Operation           :
OperationState      : COMPLETED
AutoRefreshLogOffSetting :
InHoldCustomization : False
MissingInVCenter    : False
CreateTime          : 3/3/2020 12:00:12 PM
CloneErrorMessage   :
CloneErrorTime      :
BaseImagePath        : /Datacenter_Pod1/vm/w10-1903-IC
BaseImageSnapshotPath : /VM Snapshot 3%25252f3%25252f2020, 8:22:38 AM
PendingBaseImagePath :
PendingBaseImageSnapshotPath :

PS D:\homelab> $defn.filter = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.basicState'; 'value' = 'CONNECTED' }
```

# Queries Example

```
Administrator: Windows PowerShell
CloneErrorMessage      :
CloneErrorTime         :
BaseImagePath          : /Datacenter_Pod1/vm/w10-1903-IC
BaseImageSnapshotPath  : /VM Snapshot 3%25252f3%25252f2020, 8:22:38 AM
PendingBaseImagePath   :
PendingBaseImageSnapshotPath :

PS D:\homelab> $defn.filter = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.basicState'; 'value' = 'CONNECTED' }
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> ($results.Results).data

Name      : Pod01-1004
DnsName    : pod01-1004.loft.lab
Type      : MANAGED_VIRTUAL_MACHINE
AgentVersion : 7.11.0
AccessGroup : VMware.Hv.AccessGroupId
BasicState : CONNECTED
AssignedUser :
AssignedUserName :
AgentBuildNumber : 15238678

Name      : Pod01-203
DnsName    : pod01-203.loft.lab
Type      : MANAGED_VIRTUAL_MACHINE
AgentVersion : 7.11.0
AccessGroup : VMware.Hv.AccessGroupId
BasicState : CONNECTED
AssignedUser :
AssignedUserName :
AgentBuildNumber : 15238678

PS D:\homelab> 
```

# Queries Example

```
Administrator: Windows PowerShell

PS D:\homelab> $defn.filter = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.basicState'; 'value' = 'CONNECTED' }
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> ($results.Results).data

Name           : Pod01-1004
DnsName        : pod01-1004.loft.lab
Type           : MANAGED_VIRTUAL_MACHINE
AgentVersion   : 7.11.0
AccessGroup    : VMware.Hv.AccessGroupId
BasicState     : CONNECTED
AssignedUser   :
AssignedUserName :
AgentBuildNumber : 15238678

Name           : Pod01-203
DnsName        : pod01-203.loft.lab
Type           : MANAGED_VIRTUAL_MACHINE
AgentVersion   : 7.11.0
AccessGroup    : VMware.Hv.AccessGroupId
BasicState     : CONNECTED
AssignedUser   :
AssignedUserName :
AgentBuildNumber : 15238678

PS D:\homelab> $Filter1 = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.basicState'; 'value' = 'CONNECTED' }
PS D:\homelab> $Filter2 = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.name'; 'value' = '100' }
PS D:\homelab> $filterand=new-object VMware.Hv.QueryFilterAnd
PS D:\homelab> $filterand.Filters += $filter1
PS D:\homelab> $filterand.Filters += $filter2
PS D:\homelab> $defn.filter= $filterand
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> 
```

# Queries Example

```
Administrator: Windows PowerShell
AgentBuildNumber : 15238678

Name           : Pod01-203
DnsName        : pod01-203.loft.lab
Type           : MANAGED_VIRTUAL_MACHINE
AgentVersion    : 7.11.0
AccessGroup     : VMware.Hv.AccessGroupId
BasicState      : CONNECTED
AssignedUser    :
AssignedUserName :
AgentBuildNumber : 15238678

PS D:\homelab> $Filter1 = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.basicState'; 'value' = 'CONNECTED'}
PS D:\homelab> $Filter2 = New-Object VMware.Hv.QueryFilterContains -Property @{ 'memberName'='data.name'; 'value' = '100'}
PS D:\homelab> $filterand=new-object VMware.Hv.QueryFilterAnd
PS D:\homelab> $filterand.Filters += $filter1
PS D:\homelab> $filterand.Filters += $filter2
PS D:\homelab> $defn.filter= $filterand
PS D:\homelab> $results=$queryresults = $queryservice.queryservice_create($hvservices, $defn)
PS D:\homelab> ($results.Results).data_

Name           : Pod01-1004
DnsName        : pod01-1004.loft.lab
Type           : MANAGED_VIRTUAL_MACHINE
AgentVersion    : 7.11.0
AccessGroup     : VMware.Hv.AccessGroupId
BasicState      : CONNECTED
AssignedUser    :
AssignedUserName :
AgentBuildNumber : 15238678

PS D:\homelab> 
```

# Horizon REST API

---

- Can be consumed just like any other REST api so no requirement for PowerCLI or even Powershell
- **Common Parameters**
  - Connection server url
  - Username
  - Password
  - Domain
- **Timeout**
  - Unknown to me but definitely shorter than admin console timeout.
  - Keep-Alive possible



# Horizon REST API

---

- `$Credentials = New-Object psobject -Property @{  
    username = "UserName"  
    password = "Password"  
    domain = "domain"  
}`
- `$url = "https://connectionserverFQDN"`
- `$accesstoken = invoke-restmethod -Method Post -uri "$url/rest/login" -  
    ContentType "application/json" -Body ($Credentials | ConvertTo-Json)`

# Horizon PowerCLI community Module

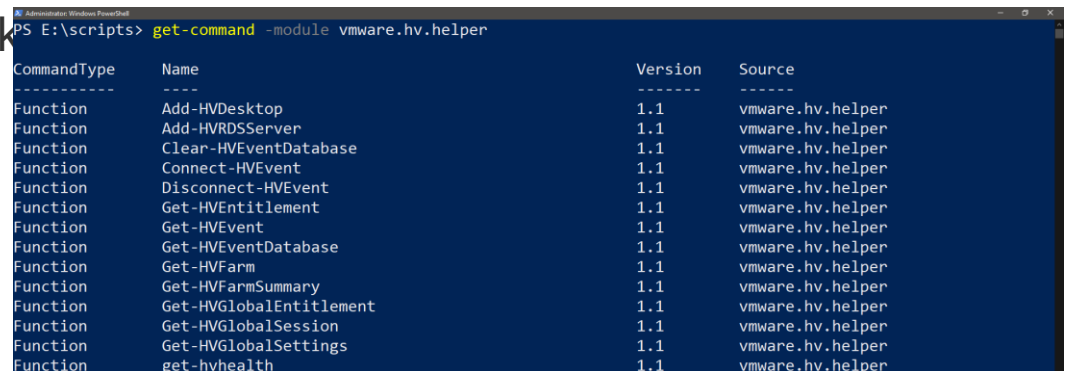
## Community module on the VMware PowerCLI Github Page

- <https://github.com/vmware/PowerCLI-Example-Scripts/tree/master/Modules/VMware.Hv.Helper>

Provides pre-created functions for more simple consumption of Horizon SOAP API's

### Examples:

- Creating & removing Desk
- Session information
- Machine information



CommandType	Name	Version	Source
Function	Add-HVDesktop	1.1	vmware.hv.helper
Function	Add-HVRDSServer	1.1	vmware.hv.helper
Function	Clear-HVEventDatabase	1.1	vmware.hv.helper
Function	Connect-HVEvent	1.1	vmware.hv.helper
Function	Disconnect-HVEvent	1.1	vmware.hv.helper
Function	Get-HVEntitlement	1.1	vmware.hv.helper
Function	Get-HVEvent	1.1	vmware.hv.helper
Function	Get-HVEventDatabase	1.1	vmware.hv.helper
Function	Get-HVFarm	1.1	vmware.hv.helper
Function	Get-HVFarmSummary	1.1	vmware.hv.helper
Function	Get-HVGlobalEntitlement	1.1	vmware.hv.helper
Function	Get-HVGlobalSession	1.1	vmware.hv.helper
Function	Get-HVGlobalSettings	1.1	vmware.hv.helper
Function	get-hvhealth	1.1	vmware.hv.helper

# More information

---

- **Horizon (SOAP) API Explorer:**

- <https://code.vmware.com/apis/956/view>
- Includes Example Code

- **Horizon (REST) API Explorer:**

- <https://code.vmware.com/home>

## VMware {Code} Slack Channel

- Sign up at: <https://code.vmware.com/join>

## My Blog

- <https://www.retouw.nl>
- Recommendation:
  - Back to Basics Series
  - Getting started with Horizon REST API